Paint House – III (View)

There is a row of `m` houses in a small city, each house must be painted with one of the `n` colors (labeled from `1` to `n`), some houses that have been painted last summer should not be painted again.

A neighborhood is a maximal group of continuous houses that are painted with the same color.

- For example: `houses = [1,2,2,3,3,2,1,1]` contains `5` neighborhoods `[{1}, {2,2}, {3,3}, {2}, {1,1}]`.

Given an array `houses`, an `m x n` matrix `cost` and an integer `target` where:

- `houses[i]`: is the color of the house `i`, and `0` if the house is not painted yet.
- `cost[i][j]`: is the cost of paint the house `i` with the color `j + 1`.

Return *the minimum cost of painting all the remaining houses in such a way that there are exactly* `target` *neighborhoods*. If it is not possible, return `-1`.

Example 1:

```
Input: houses = [0,0,0,0,0], cost = [[1,10],[10,1],[10,1],[1,10],[5,1]], m = 5, n = 2, target = 3

Output: 9

Explanation: Paint houses of this way [1,2,2,1,1]

This array contains target = 3 neighborhoods, [{1}, {2,2}, {1,1}].

Cost of paint all houses (1 + 1 + 1 + 1 + 5) = 9.
```

Example 2:

```
Input: houses = [0,2,1,2,0], cost = [[1,10],[10,1],[10,1],[1,10],[5,1]], m = 5, n = 2, target = 3

Output: 11

Explanation: Some houses are already painted, Paint the houses of this way [2,2,1,2,2]

This array contains target = 3 neighborhoods, [{2,2}, {1}, {2,2}].

Cost of paint the first and last house (10 + 1) = 11.
```

Example 3:

```
Input: houses = [3,1,2,3], cost = [[1,1,1],[1,1,1],[1,1,1],[1,1,1]], m = 4, n = 3,
target = 3

Output: -1

Explanation: Houses are already painted with a total of 4 neighborhoods
[{3},{1},{2},{3}] different of target = 3.
```

Constraints:

- `m == houses.length == cost.length`
- `n == cost[i].length`
- `1 <= m <= 100`
- `1 <= n <= 20`
- `1 <= target <= m`
- `0 <= houses[i] <= n`
- `1 <= cost[i][j] <= 10^4`